Assignment 1

Nathan Hughes (nah26@aber.ac.uk)

October 28, 2017

Contents

1 CSA Architecture

The Computer Science Alumni application aims to be relatively simple; it allows for users and posts to be created, held in a database and persist across sessions and users; This project was made following the specifications laid out in [1].

It makes ample use of the Ruby on Rails framework, using generators to quickly template large sections of code as well as a Model View Controller (Model-2 variant) to organise the overarching project.

The interaction and member variables of the two classes, User and Post can be seen in Fig.2.

1.1 User

The User class exists in order to record the information of individuals using this web application. We use the data provided here in order to link information made in the Posts class back to Users where possible.

1.2 Post

Posts is a simple class which has three attributes "title", "body" and "email". These allow for simple text based "broadcasts" to be made either anonymously or by a user, which is referenced by the "email" attribute. If the given email of the post matches with any in our Users table, then we create the link between the two.

1.3 Routes

Part of this application is having a functioning routing table, as seen in Listing.1, this allows for the route '/' of our page to be a redirection to a static homepage. Additionally an extra line has been added to redirect 404 errors back to the home page.

```
Rails.application.routes.draw do
  resources :users
  resources :posts

  root 'static_pages#home'
  # Send errors to root
  get '*path' => redirect('/')
end
```

Listing 1: Routing

1.4 Navigation

The CSA application provides a dynamic menu bar which (planned future functionality) changes the items in the menu, based on the authentication/credentials of the logged in user. Admin checking can be seen in Listing.2

```
primary.item :home, 'Home', '/home', highlights_on: /(^\/$)|(^\/home$)/
primary.item :jobs, 'Jobs', '/jobs'
primary.item :profile, 'Profile', '/profile', if: Proc.new { is_admin? }
primary.item :users, 'Users', users_path, if: Proc.new { is_admin? }
primary.item :broadcasts, 'Broadcasts', '/broadcasts', if: Proc.new { is_admin? }
```

Listing 2: Navigation items

1.5 Pagination

In order to filter for either too many users or posts on a single page Pagination has been added. This modifies the user controller (Listing.3) in order to split and give multiple selectable pages to the user, by using an additional Ruby Gem.

```ruby
  # GET /users
  # GET /users.json
  def index
    @users = User.paginate(page: params[:page],
                          per_page: params[:per_page])
                 .order('surname, forename')
  end
...
  def set_current_page
    @current_page = params[:page] || 1
  end
```

Listing 3: Modifying the user controller

1.6 Layout.scss

For a global theme a *scss* file has been used to change *css* in commonly used parts of the application. For example a snippet shown in Listing.4 shows hover and selection being given to links in the navigation bar.

```scss
.navigation {
  a {
    color: #aa1111;
    text-decoration: none;
    font-weight: bold;
    margin: 0;
    padding-left: 0.5em;
    padding-right: 0.5em;
  }

  .selected {
    background-color: #C5F1F9;
  }
  a:hover,
  a:active,
  a.here:link,
  a.here:visited {
    color: #C811D8;
  }
```

Listing 4: Layout SCSS

2 MVC Model

2.1 UML Diagrams

2.1.1 Overall class design

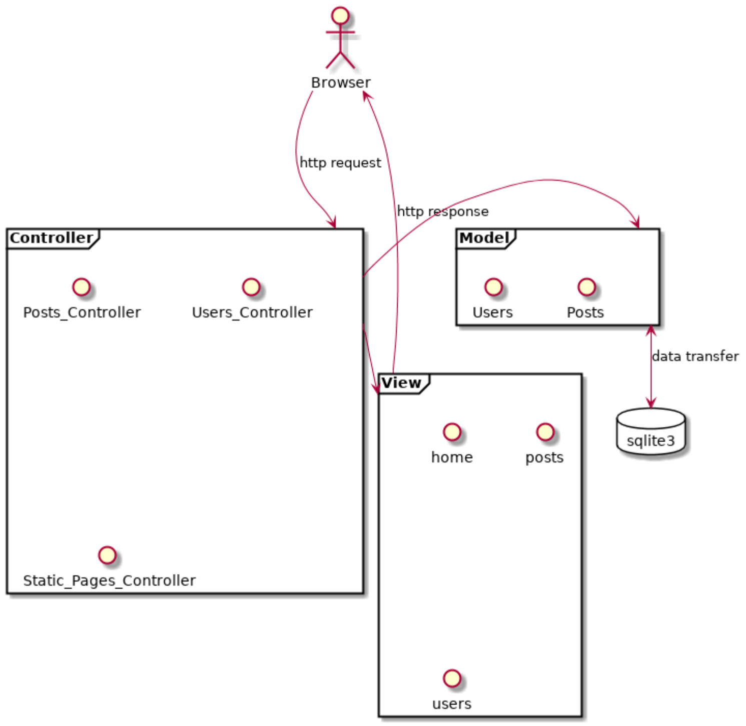

Figure 1: Posts and Users UML

2.1.2 Database

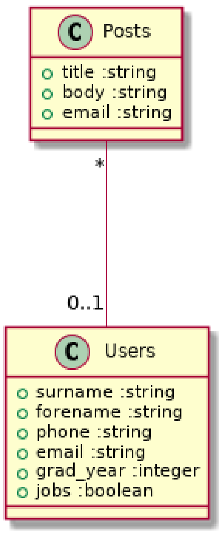

Figure 2: Posts and Users UML

2.2 What is the Model-2 Variant

The model-2 variant of the model view controller, design pattern is one designed for medium-large web-based applications [2].

In this arch-type the server acts as the controller, the view is much more separated and independent from the controller, than MVC model 1. This makes it much easier to extend the application without having decentralised components lagging behind and causing issues [3].

3 Disclaimer

Covering image used sourced from https://www.ruby-lang.org [4]. Diagrams made by myself.

Additionally I acknowledge that all information present in this document is my own work using knowledge gained from [5, 4, 3, 1, 2].

4 References

[1] Chris Loftus. Requirements/design for the cs-alumni application. *blackboard.aber.ac.uk*, 2017.

[2] Javapoint.com. Model 1 and model 2 (mvc) architecture - javatpoint. https://www.javatpoint.com/model-1-and-model-2-mvc-architecture. (Accessed on 10/28/2017).

[3] StackOverflow. java - what is the actual difference between mvc and mvc model2 - stack overflow. https://stackoverflow.com/questions/796508/what-is-the-actual-difference-between-mvc-and-mvc-model2. (Accessed on 10/28/2017).

[4] Ruby-Lang. Ruby programming language. https://www.ruby-lang.org/en/. (Accessed on 10/28/2017).

[5] Ruby on rails | a web-application framework that includes everything needed to create database-backed web applications according to the model-view-controller (mvc) pattern. http://rubyonrails.org/. (Accessed on 10/28/2017).